需求分析报告

第三大组第三小组

2024 年 4 月 7 日

目录

第一部分：引言

1.1 编写目的

随着移动互联网、人工智能、大数据等技术快速发展，越来越多的人享受到互联网医疗带来的便利服务。例如，在线开具电子处方，药品直接快递到家；远程诊疗，在乡镇医院也能享受省级专家会诊；开通电子社保卡，动动手指就能挂号缴费。互联网医疗借助防疫期间庞大的用户需求，实现迅猛发展，并不断向潜在用户渗透。我们希望编写一个综合性的互联网在线医疗平台，为病人、医生和院方提供流程和管理上的便利。

1.2 相关背景

该项目开发的软件为一个综合性的互联网在线医疗平台，旨在为病人、医生和院方提供便利的医疗服务。该平台将采用前后端分离的架构，前端使用 Vue3 框架实现用户界面，后端使用 Spring Boot 框架开发服务器端逻辑，数据库采用 MySQL 关系型数据库。相较于传统的“纸质 + 线下”模式而言，该项目的“互联网 +”医疗模式具有便于归档的优势，并且可以使得医疗资源匮乏的地区也能享受到高质量的医疗服务。该平台将整合医院的资源，提供在线挂号、预约、缴费、问诊等服务，让病人、医生和管理人员足不出户便能完成全过程的在线问诊。该平台将采用多种信息安全技术保护用户信息和交易安全。该平台将遵循便民、利民的理念，简化操作流程、美化操作界面、便利各方用户的使用。

平台上线后，病人可以通过平台进行挂号、预约、缴费、问诊等操作，医生可以通过平台进行打卡、上班、请假、查询等操作，管理人员可以通过平台进行药房管理、医生管理、病人管理、投诉管理等操作。

本小组负责开发的子系统实现了线上问诊与投诉功能，以及管理端的投诉处理功能。

1.3 定义部分

1.3.1 Linux

Linux 是一种自由和开放源码的类 UNIX 操作系统。该操作系统的内核由 Linus • Torvalds 在 1991 年 10 月 5 日首次发布。Ubuntu 22.04 LTS 是 Canonical 于 2022 年 4 月 21 日发布的 Linux 发行版之一。本项目运行在云端 Ubuntu 操作系统的服务器上。

1.3.2 MySql

MySQL 是一个关系型数据库管理系统，由瑞典 MySQL AB 公司开发，属于 Oracle 旗下产品。MySQL 是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL 是最好的 RDBMS 应用软件之一。本项目采用 MySQL 作为数据库管理系统。

1.3.3 SpringBoot

Spring Boot 是由 Pivotal 团队提供的全新框架，其设计目的是用来简化新 Spring 应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。本项目采用 Spring Boot 作为后端开发的框架。

1.3.4 Vue3

Vue 是一款用于构建用户界面的 JavaScript 框架。它基于标准 HTML、CSS 和 JavaScript 构建，并提供了一套声明式的、组件化的编程模型，帮助你高效地开发用户界面。无论是简单还是复杂的界面，Vue 都可以胜任。本项目采用 Vue3 作为前端开发的框架。

1.3.5 用例

用例（use case），是软件工程或系统工程中对系统如何反应外界请求的描述，是一种通过用户的使用场景来获取需求的技术。每个用例提供了一个或多个场景，该场景说明了系统是如何和最终用户或其它系统互动，也就是谁可以用系统做什么，从而获得一个明确的业务目标。

1.3.6 用例图

用例图（use case diagram）是用户与系统交互的最简表示形式，展现了用户和与他相关的用例之间的关系。通过用例图，人们可以获知系统不同种类的用户和用例。用例图也经常和其他图表配合使用。

用例图的目的就是为了可以让人在一个更高的层次概览整个系统，用平白的话语让项目参与者理解系统。它可以辅以额外的图表和文档，以更加完整地展现系统的功能和技术细节。

1.3.7 状态图

状态图（Statechart Diagram）是 UML 的五个图之一，用来模拟系统的动态性质。它们定义了一个对象在其生命周期中的不同状态，这些状态由事件改变。状态图对于建模反应式系统很有用。反应式系统可以被定义为对外部或内部事件做出反应的系统。

状态图描述了从一个状态到另一个状态的控制流。状态被定义为一个对象存在的条件，当一些事件被触发时，它就会改变。状态图最重要的目的是对一个对象从创建到终止的生命周期进行建模。

1.3.8 数据流图

数据流图（Data Flow Diagram）是描述系统中数据流程的一种图形工具，它标志了一个系统的逻辑输入和逻辑输出，以及把逻辑输入转换逻辑输出所需的加工处理。

数据流图不是传统的流程图或框图，数据流也不是控制流。数据流图是从数据的角度来描述一个系统，而框图是从对数据进行加工的工作人员的角度来描述系统。

1.3.9 CRC 卡

CRC 卡（Class-Responsibility-Collaborator）是一种在面向对象分析建模中常用的工具。这个方法在面向对象工程设计中非常流行。用户、设计者和开发人员都参与其中，完成对整个面向对象系统的设计。它包括三个部分：类名、类的职责和类的协作。CRC 卡是一个标准索引卡集合，每张卡片表示一个类。这种方法帮助我们更好地理解系统的结构，确保设计的高质量和可维护性。

1.3.10 类图

类图（Class diagram）是软件工程的统一建模语言一种静态结构图，该图描述了系统的类集合，类的属性和类之间的关系。类图是面向对象式的建模。他们一般都被用于概念建模（conceptual modelling）的系统分类的应用程序，并可将模型建模转译成代码。

第二部分：任务概述

2.1 系统概述

本小组负责的问诊投诉子系统是整个医疗系统的一个重要组成部分，提供了病人和医生之间的在线问诊服务，以及病人对医生和医院的投诉功能。医生与病人可以通过系统进行在线问诊，病人可以向医生咨询病情、诊断结果、治疗方案等问题，医生可以根据病人的症状和病史进行诊断和治疗建议。同时，病人还可以对医生和医院的服务进行评价和投诉，以提高医疗服务的质量和效率。问诊投诉子系统的目标是提供便捷、高效、安全的在线问诊服务，为病人和医生提供更好的医疗体验。

2.2 系统组成

本系统的架构包括前端界面、后端服务器和数据库：
前端界面：用户通过 Web 页面或移动应用程序与系统进行交互。
后端服务器：包括用户管理、医生管理等模块，处理用户请求并与数据库交互。
数据库：存储用户信息、医疗记录和系统配置等数据。

技术实现：系统采用前后端分离的架构，前端使用 HTML、CSS 和 JavaScript 并采取 Vue3 框架实现用户界面，后端使用 Java 语言和 SpringBoot 框架开发服务器端逻辑，数据库采用 MySQL 关系型或非关系型数据库。同时采取多种信息安全技术保护用户信息和交易安全。

2.3 用户特点

- 病人用户：病人用户是有诊疗需求的个人。他们希望得到高质量的线上医疗服务。
- 医生用户：医生用户是注册在系统中提供医疗服务的医护人员。具有丰富的医疗知识和经验，负责为病人提供诊断、治疗建议和处方。
- 管理员用户：平台管理员拥有医疗平台的最高权限，负责平台药品库存情况、投诉处理、绩效考核以及诊疗安排。此外，管理员掌控病人、医生和医院的所有信息，需要监督系统的安全性和稳定性，确保用户数据的保密性和完整性。

2.4 假定和约束

1. 本系统假定，背景是在一个物联网高度发达的环境下。
2. 本系统假定三方使用者都是在家然后线上在线完成全过程的。
3. 本系统假定例如血常规等检测不用去医院，在家附近就有智能的检测站。
4. 并且检测的结果是全国的医疗系统都共同可以查询的。
5. 同时药品采用邮寄的形式寄送到家中。
6. 本系统假定病人一天内的所有活动同属一张病历单，不存在一天看两种不同病的可能性。
7. 本系统规定病人第一天和第二天的看病不属于一张病历单。

2.5 子系统功能

产品使用者可分为上述的 3 种用户，依据各个用户拥有的权限，问诊投诉子系统的功能集中于以下几个方面：

表 1: 子系统功能

功能	设计的用户类别
投诉	病人、管理员、医生
问诊	病人、医生
后台数据管理	管理员
开药/检测	病人、医生

2.6 设计和实现的约束

系统的设计、编码、以及维护将遵照所提交的《软件需求报告》等文档进行。在只能医疗平台具体的设计和实现上，按照以下约束进行：

- 用户体验：线上医疗平台的成功与否在很大程度上取决于用户体验。因此，设计和实现平台时必须将用户体验作为重要因素考虑。平台必须易于使用、直观且流畅。
- 数据存储：项目产品使用标准 Mysql 数据库系统作为引擎，按照数据产生、转换和存储的策略，通过将数据导入数据库的方式进行数据的存储操作。
- 数据隐私：在线上交易中，用户的数据隐私非常重要。因此，在设计和实现平台时必须保证用户的个人信息得到保护，并采取必要的措施来防止数据泄露。
- 系统可靠性：平台必须具备高可靠性和稳定性。任何系统中都可能出现故障，但平台必须具备快速恢复的能力，以避免交易中断和用户流失。
- 法律和合规要求：任何一家医院在运营过程中都必须符合法律和合规要求。线上医疗平台也不例外。在设计和实现平台时，必须遵守当地的法律和法规，特别是与医疗行业相关的法律和法规。
- 网络服务吞吐：根据项目要求，本项目要求提供远程服务的能力，以确保同时为至少 10000 名用户进行线上问诊的要求。

2.7 用户文档

产品交付将为用户提供三类文档：描述类文档、过程类文档、参考类文档，主要帮助用户可以快速上手商品交易系统网站，并在遇到实际问题时可以通过文档查阅快速解决所遇到的问题。

1. 描述类文档：描述类文档提供对于线上医疗问诊投诉系统网站基本组成、属性、功能、特性、接口、应用的描述信息，用于帮助用户概览问诊投诉系统网站所具备的所有功能以及各个功能的具体使用方式。

2. 过程类文档：过程类文档实际上通过用户在第一次登录系统时以及第一次使用某种功能时进行呈现，通过指引式的教学环节设计使用户对于各个功能的具体使用流程有基本而具体的了解。

3. 参考类文档：参考类文档按照专题提供信息，用于为用户提供在进行问诊投诉系统网站中某种操作以及理解其中某项功能时所需要的详细记录以及解释，同时为用户提供问题的快速解决方案，以便于用户进行操作。

2.8 术语说明

- 软件：一系列按照特定顺序组织的计算机数据和指令的集合。
- 软件工程：是 (1) 将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件；(2) 对在 (1) 中所述方法所进行的研究。
- 软件质量：软件与明确的和隐含的定义的需求相一致的程度。
- 质量认证：也叫合格评定，是国际上通行的管理产品质量的有效方法。
- 软件过程：一个为建造高质量软件所需完成的任务的框架，即形成软件产品的一系列步骤，包括中间产品、资源、角色及过程中采取的方法、工具等范畴。
- 软件需求：(1) 用户解决问题或达到目标所需条件或权能 (Capability)。(2) 系统或系统部件要满足合同、标准、规范或其它正式规定文档所需具有的条件或权能。(3) 一种反映上面 (1) 或 (2) 所述条件或权能的文档说明。它包括功能性需求及非功能性需求，非功能性需求对设计和实现提出了限制，比如性能需求、质量标准或者设计限制。
- 业务需求：反映组织机构或客户对系统或产品高层次的目标要求，它们在项目视图与范围文档中予以说明。
- 用户需求：描述了用户使用产品必须要完成的任务，可以在用例模型或方案脚本中予以说明。
- 功能需求：定义了开发人员必须实现的软件功能，使得用户能完成他们的任务，从而满足了业务需求。
- 非功能需求：从各个角度对系统的约束和限制，反映了应用对软件系统质量和软件需求规格说明书，用于反映商品交易系统网站的额外需求。
- 需求工程：指应用已证实有效的技术、方法进行需求分析，确定客户需求，帮助分析人员理解问题并定义目标系统的所有外部特征的一门学科。它通过合适的工具和记号系统地描述待开发系统及其行为特征和相关约束，形成需求文档，并对用户不断变化的需求演进给予支持。

- 用例图：指由参与者、用例以及它们之间的关系构成的用于描述系统功能的静态视图。
- 项目管理：通过合理地组织和利用一切可以利用的资源，按照计划的成本和计划的进度，完成一个计划的目标，它包含团队管理、风险管理、采购管理、流程管理、时间管理、成本管理和质量管理等。
- ISO9000：一类标准的统称，由质量管理体系技术委员会所制定的所有国际标准。

第三部分：需求规定

3.1 功能要求

(1) 基本功能：系统能够执行其基本预期的功能，满足用户最基本的需求。
(2) 数据处理功能：系统能够有效地处理用户输入的数据，并根据需要进行存储、检索、修改和删除等操作。数据处理功能应该准确、可靠，并保证数据的完整性和安全性。
(3) 安全功能：系统应该具有必要的安全功能，包括身份认证、权限控制、数据加密等。系统应该能够防范各种安全威胁，如 SQL 注入、跨站脚本攻击等。
(4) 可扩展性功能：系统应该具有良好的可扩展性，能够方便地进行功能扩展和模块添加。系统应该支持插件化或模块化的设计，以便将来根据需要进行功能的增加和修改。
(5) 可维护性功能：系统应该易于维护和管理，包括代码结构清晰、注释充分、文档完备等。系统应该具有良好的可读性和可维护性，便于开发团队进行后续的维护和修改。

3.2 用户需求

1. 患者
   (a) 选择对应的医生，进入问诊
   (b) 以文字或语音方式向医生描述自己的病症
   (c) 使用移动设备拍摄或直接上传患处图片或视频
   (d) 对医生进行评价或投诉
2. 医生
   (a) 查看患者医疗档案
   (b) 与患者以文字或语音方式进行交流
   (c) 要求病人进入对应科室进行血常规、影像等各项检查
   (d) 查看病人化验或检查结果
   (e) 开具药品处方
   (f) 安排复诊

(g) 查看病人评价、投诉情况

3. 管理员

(a) 查看各科室问诊情况

(b) 查看患者评价、投诉

(c) 向医生反馈病人评价、投诉

3.3 性能要求

(1) 响应时间要求：

系统对用户请求的响应时间要求，在用户发起请求后系统应尽快返回结果，以保证用户体验。

(2) 吞吐量要求：

系统应在单位时间内能够处理的请求或事务数量的要求，通常用于衡量系统的处理能力。

(3) 并发性能要求：

系统应能够同时处理的用户数量或并发请求数量的要求，涉及系统的并发处理能力。

(4) 容错性能要求：

系统应在出现性能问题或部分故障时有容错处理能力，保障系统的稳定性和可靠性。

(5) 资源利用效率要求：

系统应能有效地利用计算资源、存储资源和网络带宽，以提高系统的性能。

(6) 可用时间性能要求：

系统应有可用时间性能要求，即系统在一定时间内可用的比例，通常以百分比表示。

3.4 接口要求

首先，接口应该被清晰地定义和文档化，包括输入参数、输出参数、返回值类型、异常处理等内容，以便开发人员正确地使用接口。其次，接口应该保证稳定性、一致性、安全性、可用性，同时易于使用和理解，具有良好的可读性和可维护性。

3.5 界面要求

首先，界面应该设计得简洁明了，易于理解和操作。同时，使用直观的图标、文字和布局，提供统一的视觉风格和操作方式。界面应该具有良好的响应性，快速响应用户的操作，且能够友好地处理用户输入错误或操作失误。加之提供及时的交互反馈，如动画效果、进度条等。

第四部分：用户场景

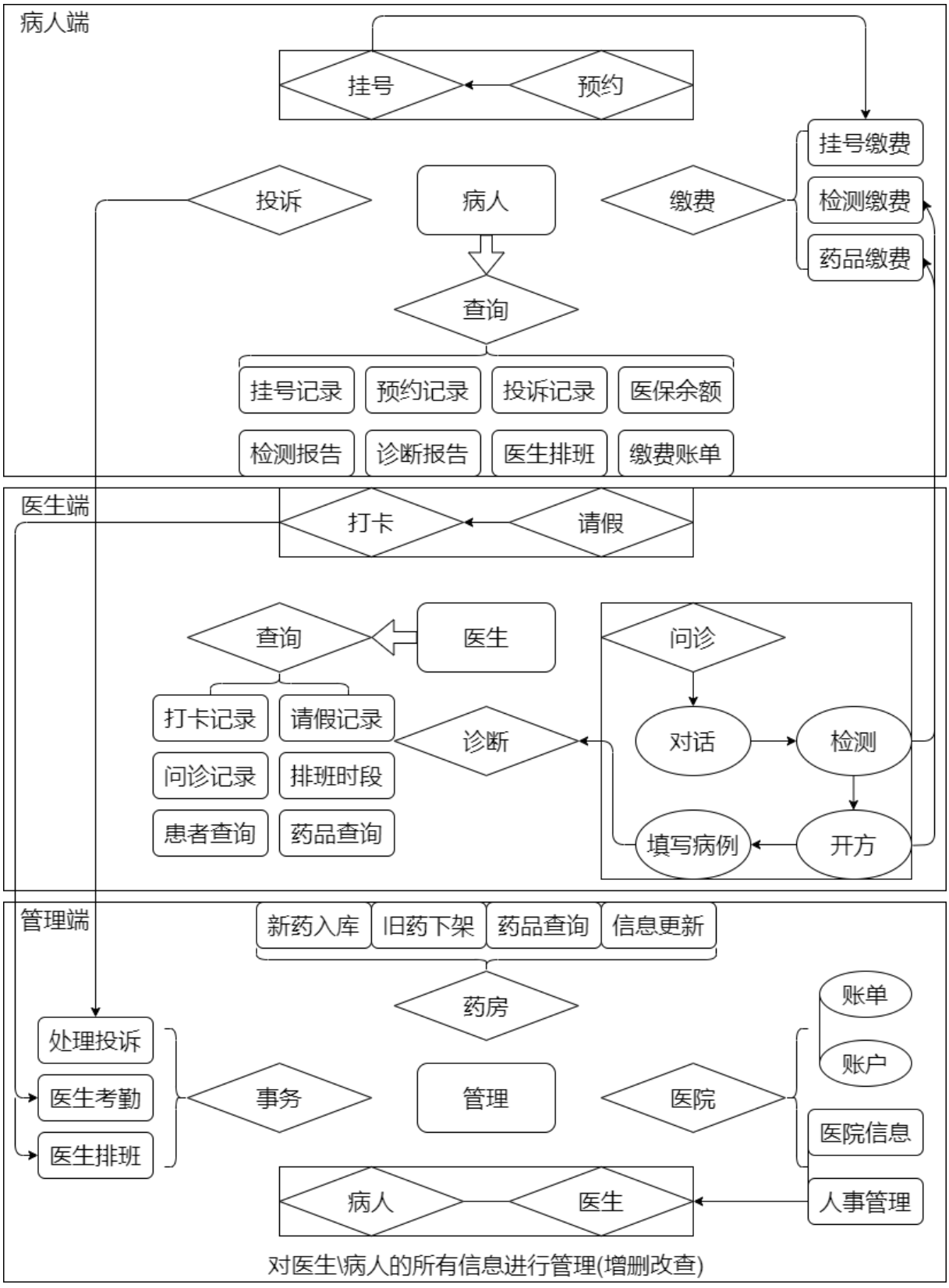


图 1. 总架构图

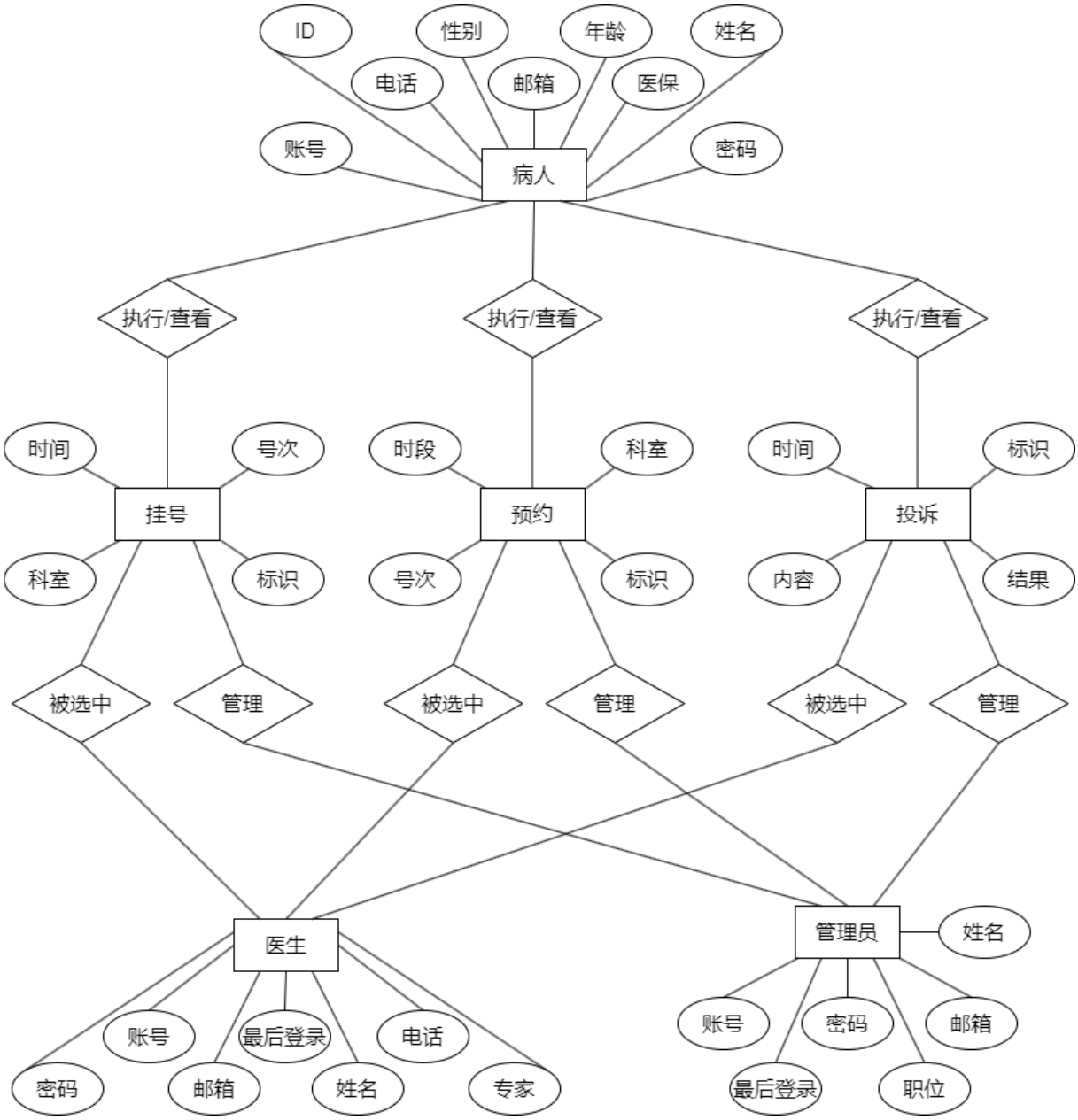


图 2.E-R 图 (1)

这张是病人的一些基础功能的 E-R 图，包括挂号、预约和投诉与各个实体之间的关系。

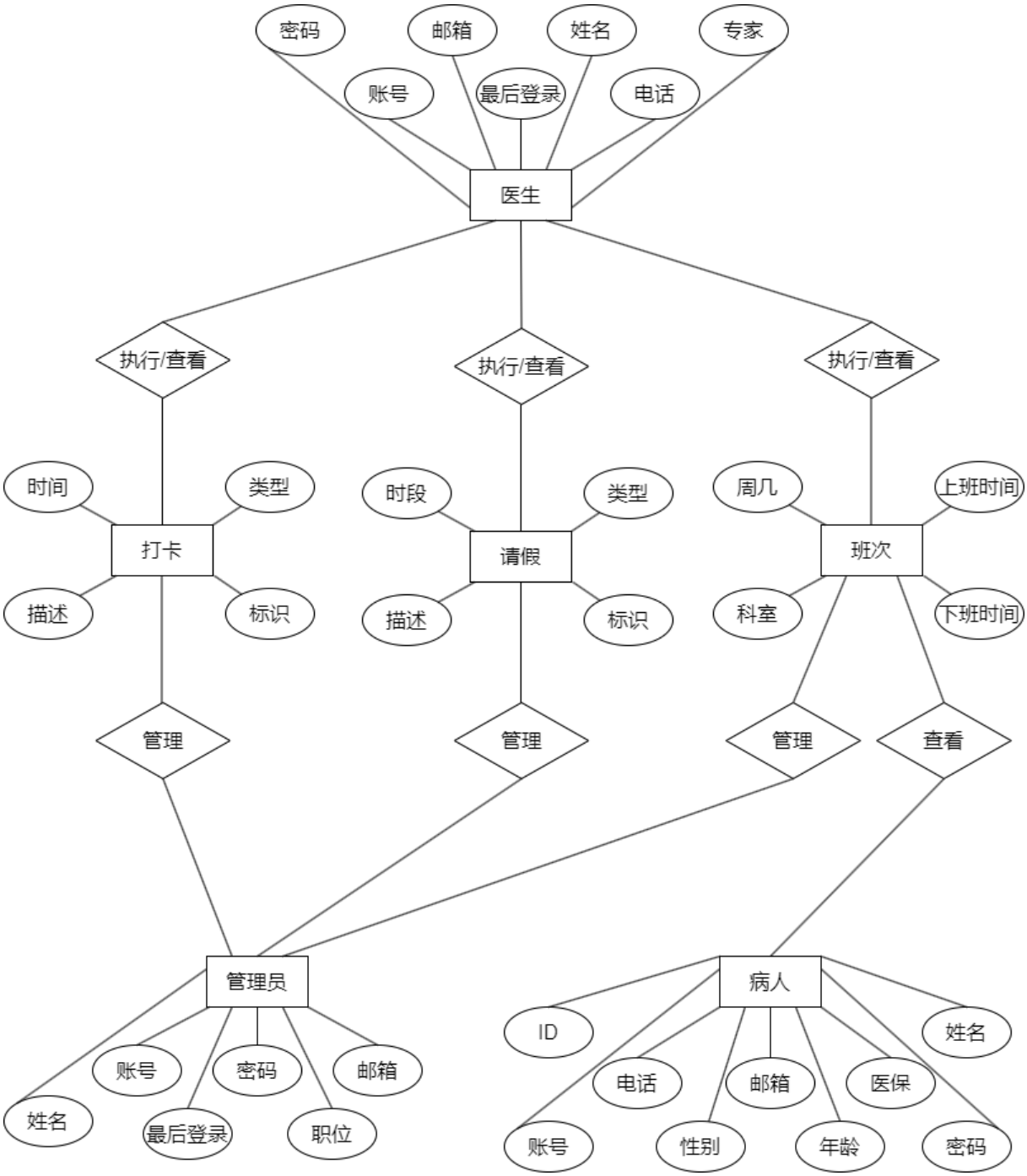


图 3.E-R 图 (2)

这张是医生的一些基础功能的 E-R 图，包括打卡、请假和排班与各个实体之间的关系。

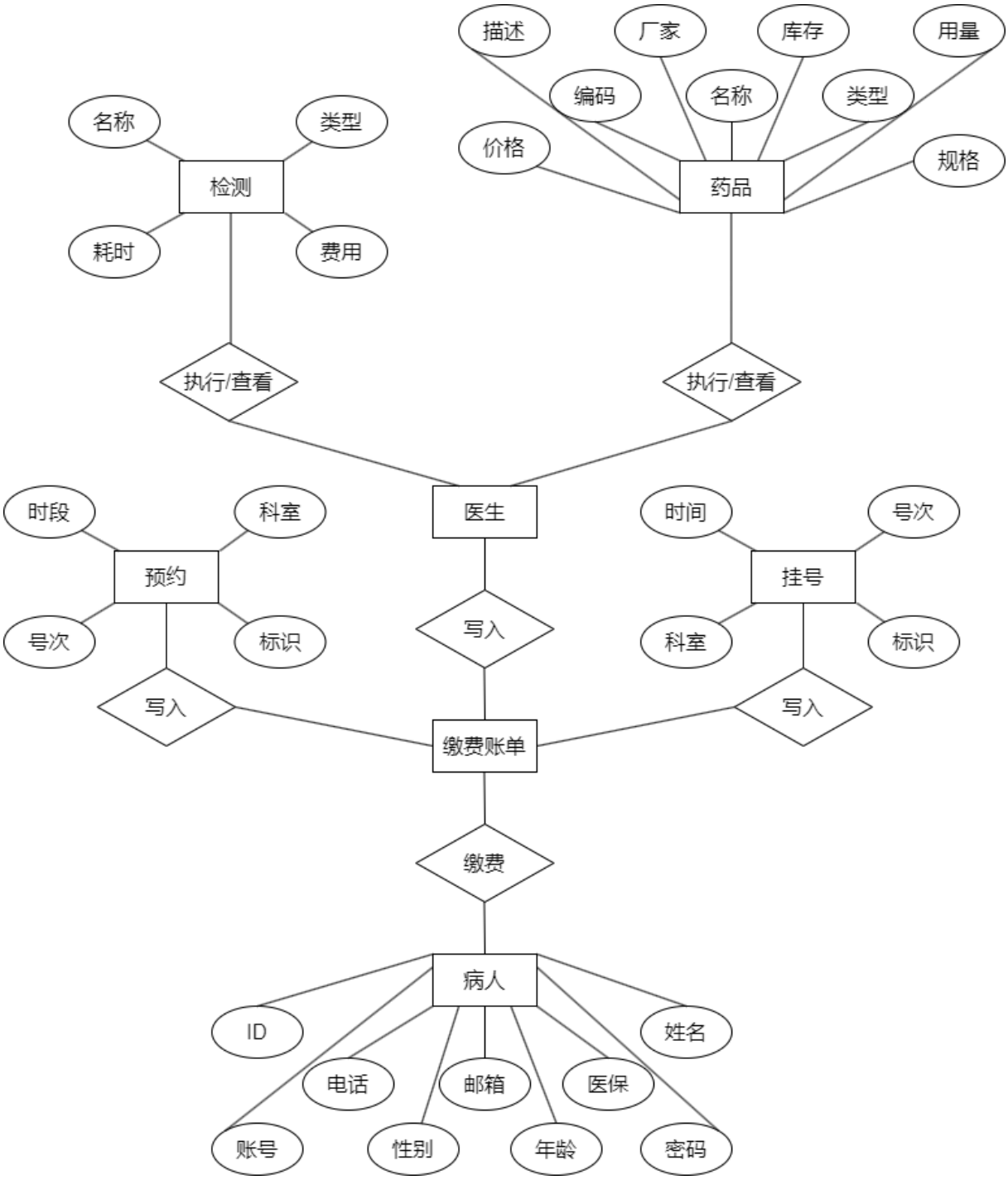


图 4.E-R 图 (3)

这张是用来描述医生与检测和开方以及病人与缴费各个实体之间关系的 E-R 图。

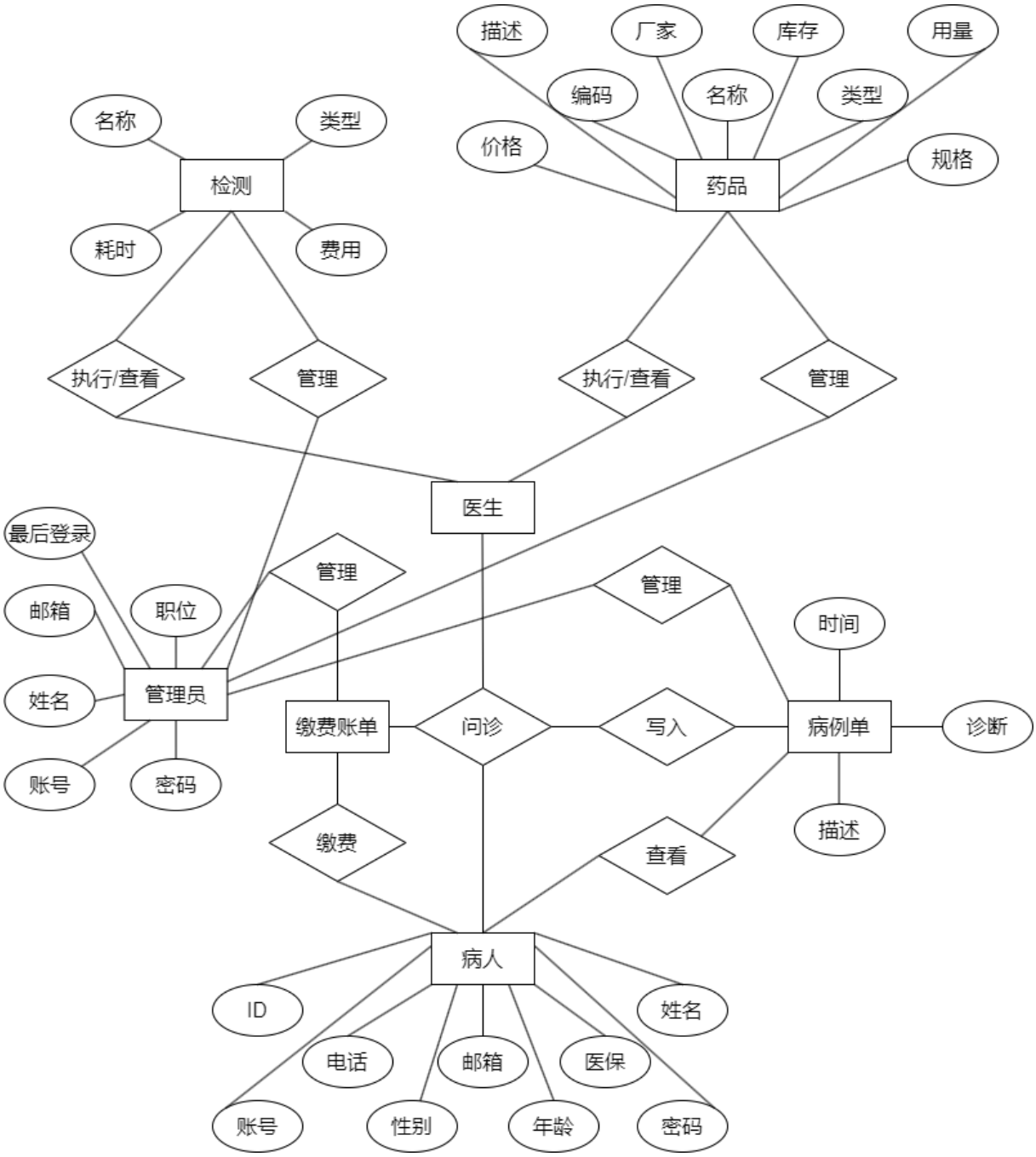


图 5.E-R 图 (4)

这张是用来重点描述病人医生问诊时各个实体之间关系的 E-R 图。

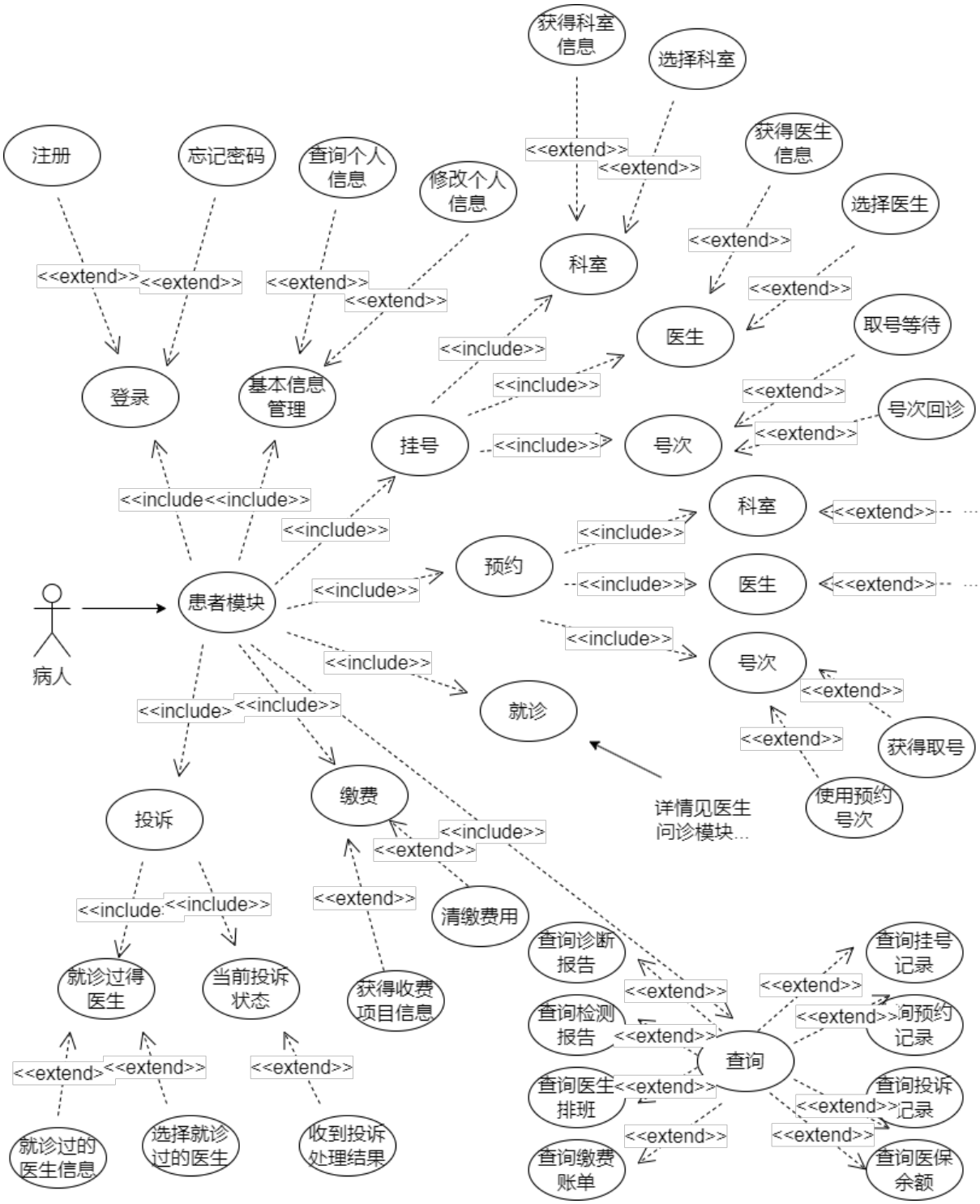


图 6. 用例图 (1)

病人端的总用例图。

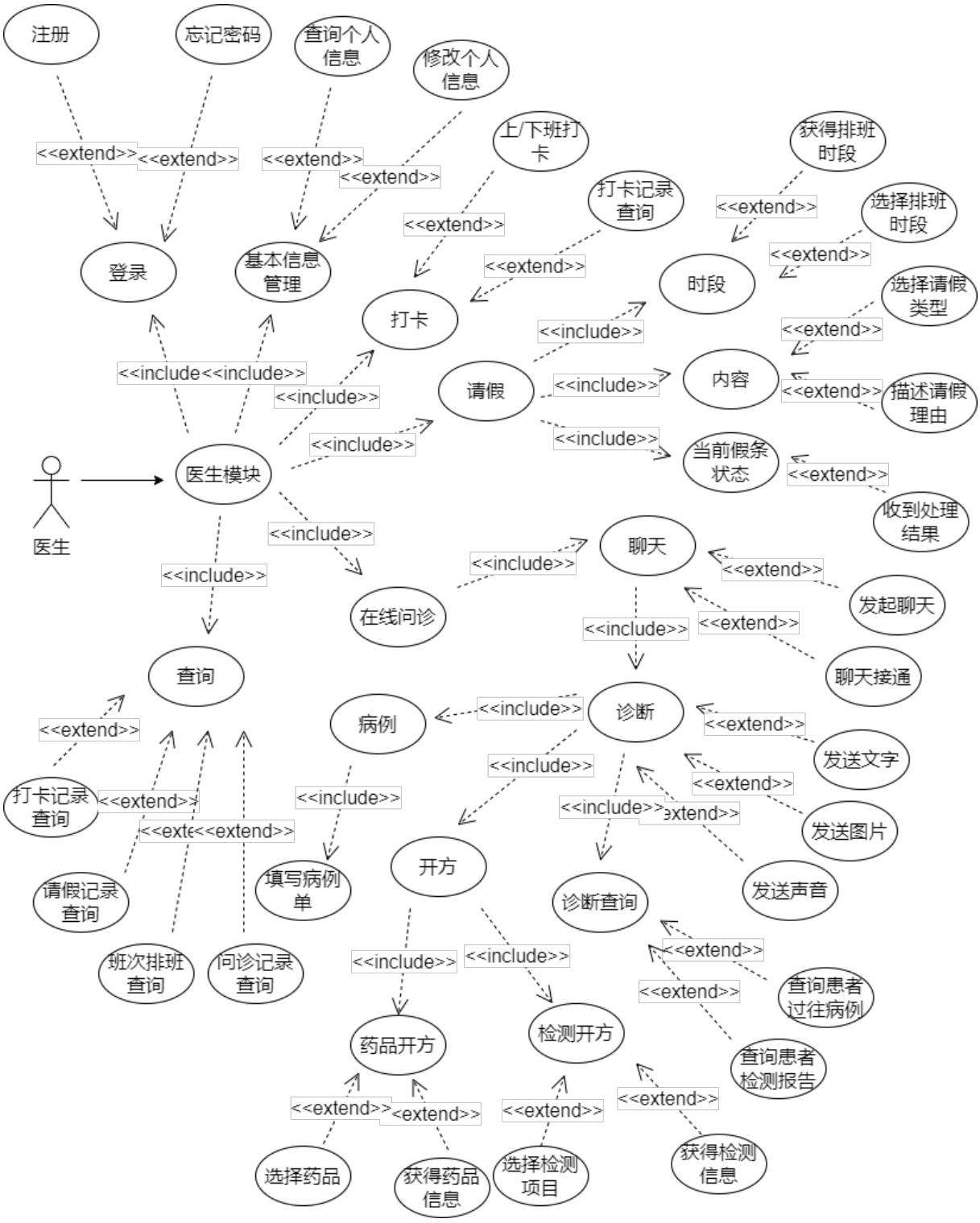


图 7. 用例图 (2)

医生端的总用例图。

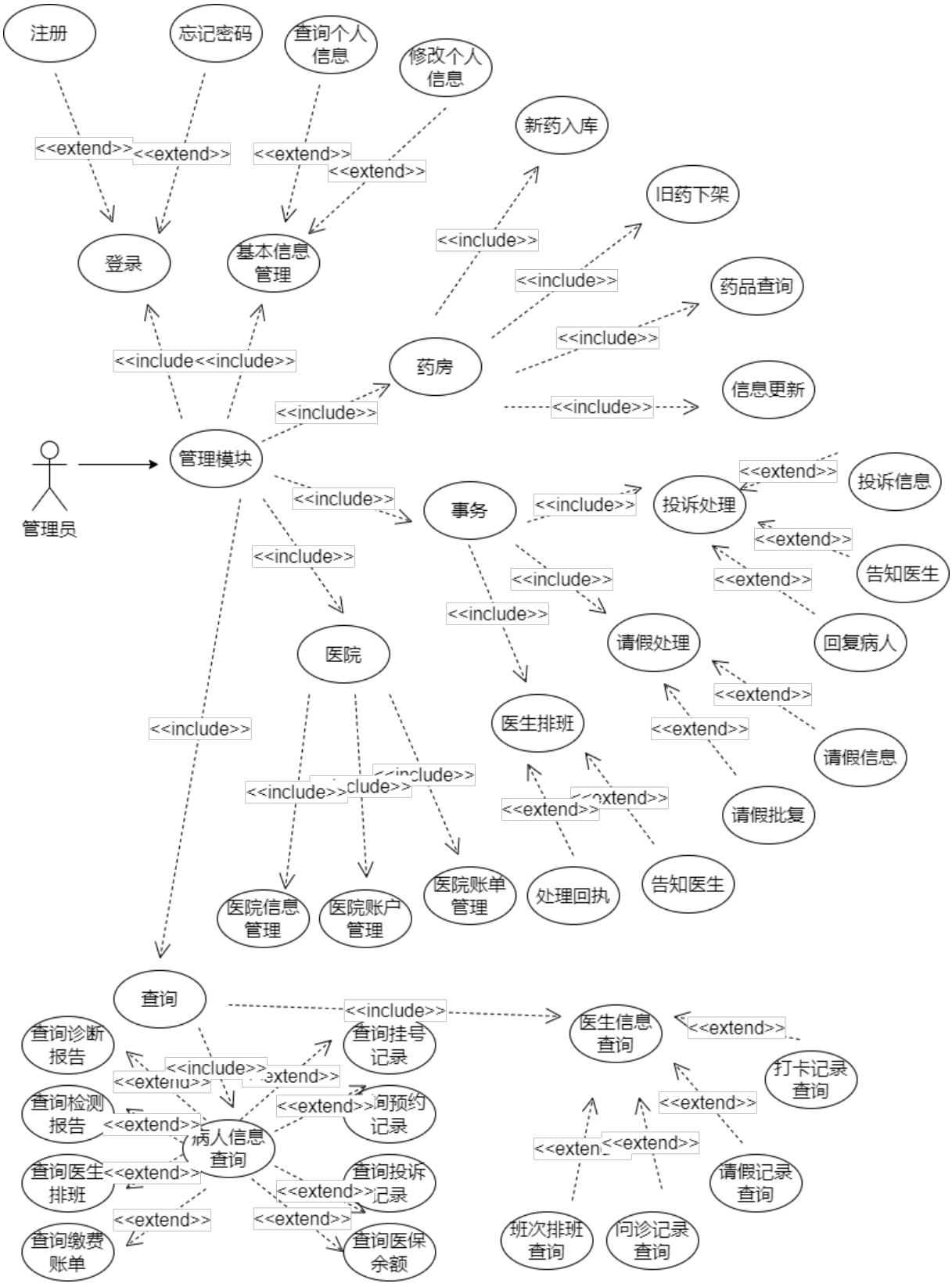


图 8. 用例图 (3)

管理端的总用例图。

	对象	行为
01	病人、医生、管理	登录
02	病人、医生、管理	个人信息管理（修改、查询）
03	病人	挂号、预约
04	病人	缴费（挂号、检测、药品缴费）
05	病人、管理	投诉（处理投诉）
06	医生、病人	诊断（问诊）
07	医生、管理	打卡、请假（处理请假、排班）
08	医生、病人	查询（各种信息的查询）
09	管理	药房管理
10	管理	医院、医生、病人所有信息的管理

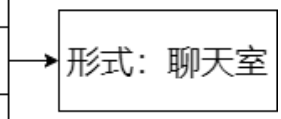


图 9. 分工图

将上面的总体架构图、E-R 图和用例图进行具体的细化分工之后，就得到了上面的这张分工图。分工图将整个系统分成具体的 10 个功能模块，其中

•）01、02 由本小组（第二小组完成）；

•）03、04 由第一小组完成；

•）05、06 由第三小组完成；

•）07、08 由第四小组完成；

•）09、10 由第五小组完成。

并且每个功能模块包含

(1)E-R 图、(2) 用例图、

(3) 流程图、(4) 数据流图、

(5) 类图、(6) 状态图、

(7) 时序图、(8)CRC 卡。

并附上这些图片的描述和解释。

接下来是本小组部分的图解：

4.1E-R 图

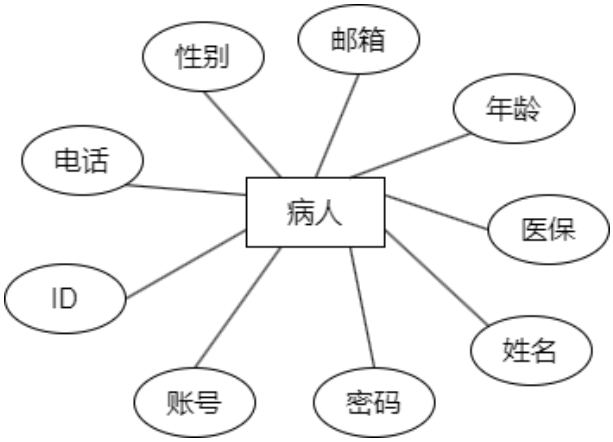


图 1.E-R 图示例

4.2 用例图

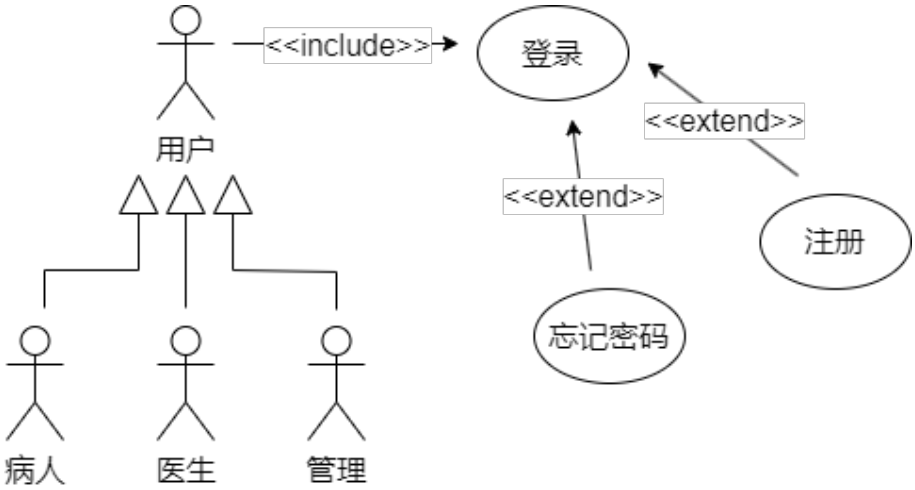


图 2. 用例图示例

其中，三角形箭头表示泛化关系。include 箭头表示基础功能，extend 箭头表示拓展功能。

4.3 流程图

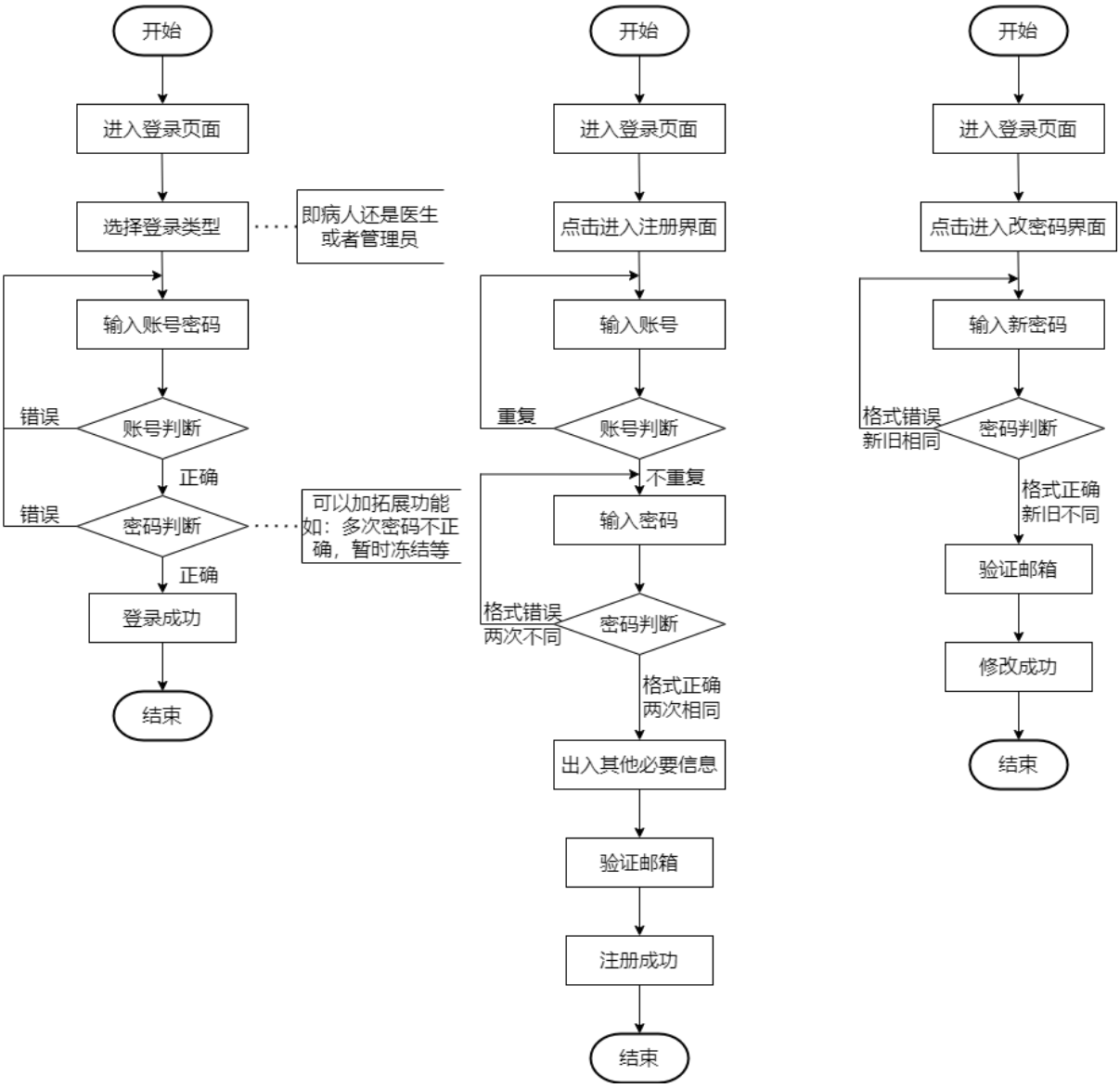


图 3. 流程图示例

其中，菱形为判断分支，虚线加三边框为注解。

4.4 数据流图

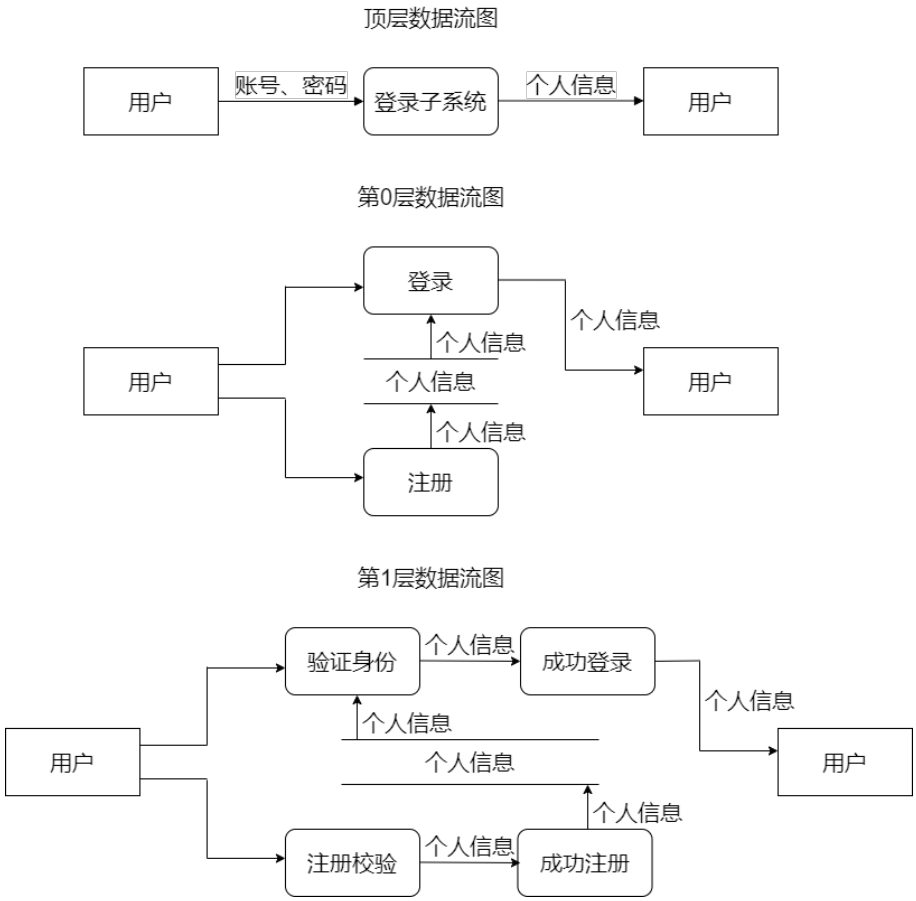


图 4. 数据流图示例

其中，箭头代表数据流动的方向，箭头上注明数据的内容。方框代表开始和结束，圆角框代表数据处理的过程，两行则是数据存储。

4.5 类图

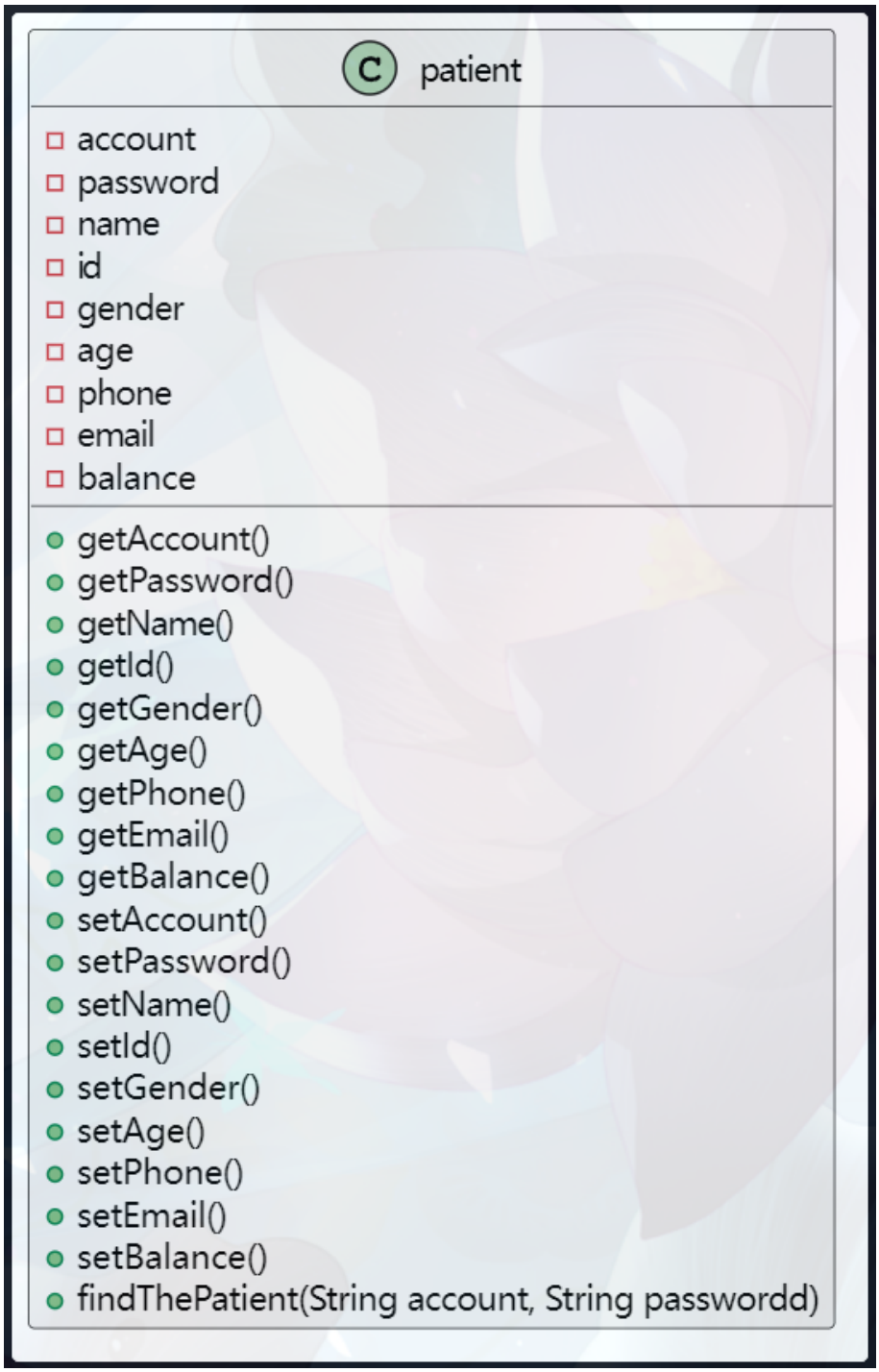


图 5. 类图示例

其中，红框代表 private，绿圆代表 public。

4.6 状态图

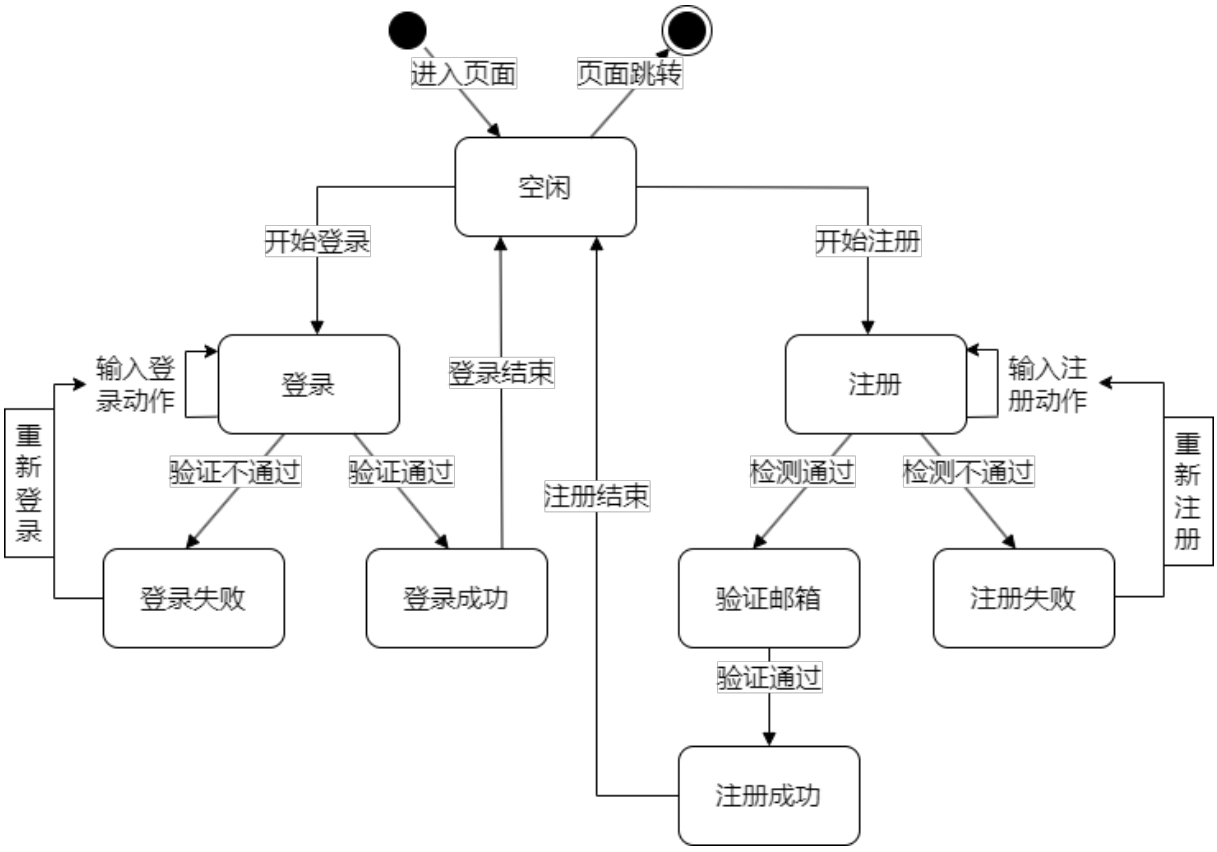


图 7. 状态图示例

其中，大圆点为开始，外面加一个环为结束。

4.7 时序图

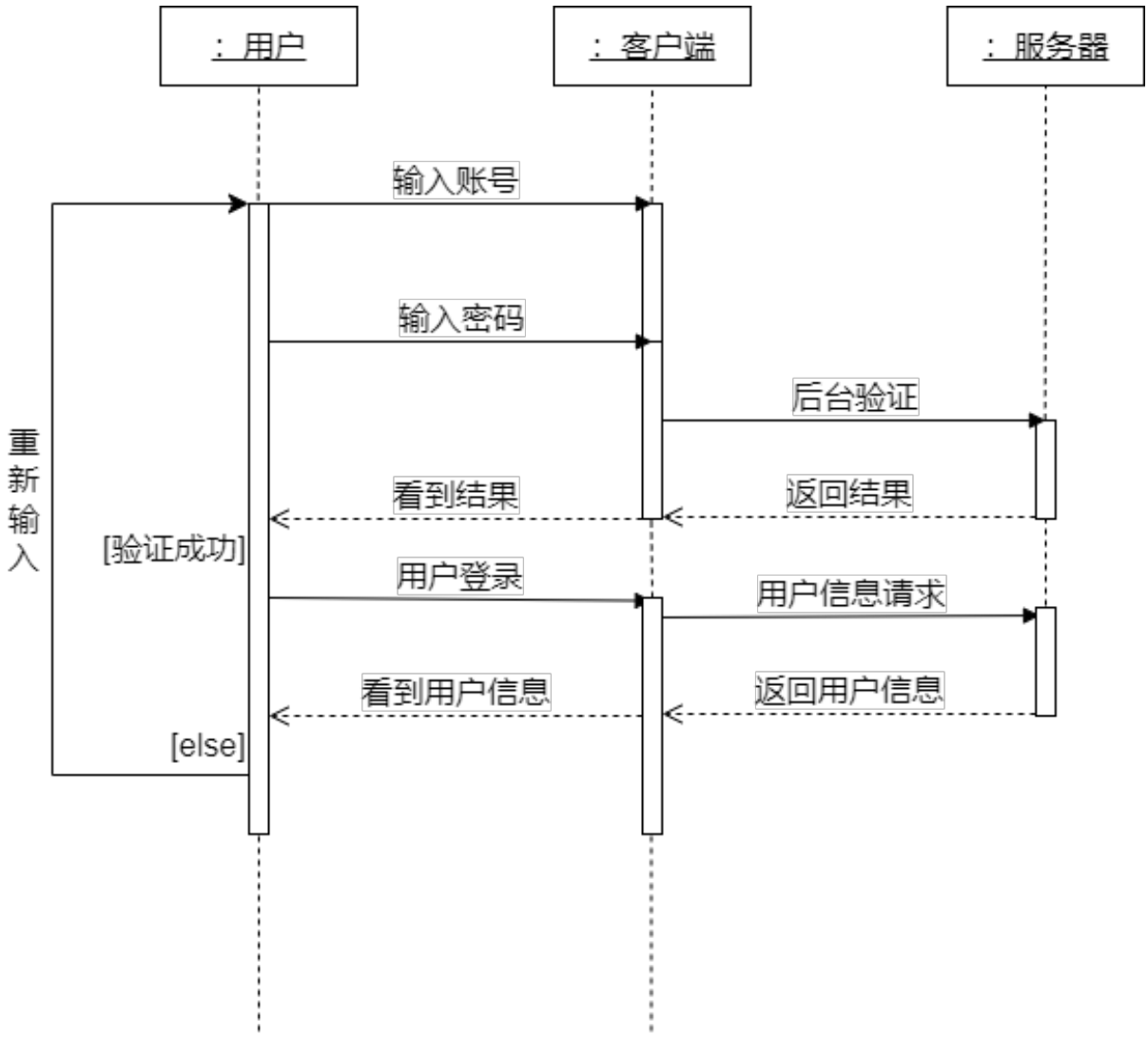


图 7. 顺序图示例

其中，竖着的圆柱条代表生命线。

4.8 CRC 卡

class：患者类	
说明：完成一次登录	
职责：	协作类
返回患者信息	无

图 8.CRC 卡示例

第五部分：运行环境

5.1 服务端软硬件配置需求

硬件配置：

1. 至少 1 核 CPU，使用 Intel Xeon 或类似的服务器级处理器。

2. 内存至少 4GB，以支持并发请求和数据库操作。

3. 存储：至少 50GB 的高速固态硬盘（SSD），以确保快速的数据读写操作。

软件配置：

1. 操作系统：Linux（例如 Ubuntu Server、CentOS）或类 Unix 系统，具有稳定性和安全性。

2. 数据库：使用高性能的关系型数据库，MySQL 以存储用户数据、医疗记录和系统配置。

3. 后端框架：选择 Spring Boot 作为开发团队的后端框架。

4.Web 服务器：选择 Apache Tocmat 服务器，用于处理 HTTP 请求并提供静态文件服务。

5. 安全性：有适当的安全措施，包括防火墙、数据备份和恢复机制等。

5.2 客户端软硬件配置需求

硬件配置：

PC 端：建议使用具有至少 4GB RAM 和双核 CPU 的计算机，以保证流畅的系统运行。

移动端（暂定）：支持 iOS 和 Android 平台的智能手机和平板电脑，具有足够的处理能力和内存，以便用户能够顺畅地使用您的移动应用。

软件配置：

操作系统：PC 端可以支持 Windows、macOS 或 Linux 操作系统；移动端需要 iOS 10 或更新版本，或者 Android 5.0 或更新版本。

浏览器：对于 Web 应用程序，支持常见的现代浏览器，包括 Chrome、Firefox、Safari 和 Edge 等主流浏览器版本。

5.3 需求变更规范

(1) 提出需求变更：

•）用户或相关利益相关者应以书面形式提出需求变更请求，明确描述变更内容和理由。

•）变更请求应提交给指定的变更管理团队或项目经理。

(2) 评估变更影响：

•）变更管理团队应及时评估变更对项目进度、成本和资源的影响，并与相关者讨论确认。

•）评估还应考虑变更对系统功能、质量和安全性的影响。

(3) 变更批准：

•）只有经过适当的评估和审批后，变更才能被批准。

•）变更批准通常由项目管理委员会或项目经理负责。

(4) 实施变更：

•）一旦变更被批准，变更管理团队应与开发团队协作，确保变更被正确实施。

•）变更实施后，需要进行必要的测试和验证，以确保系统功能的正确性和稳定性。

5.4 故障处理规范

(1) 故障报告：

•）用户或系统监控应用程序应能够报告系统故障或异常。

•）故障报告应包括故障描述、发生时间、影响范围等相关信息。

(2) 故障诊断：

•）技术支持团队应及时响应故障报告，并进行故障诊断。

•）诊断过程可能涉及查看日志、分析系统状态和运行情况等。

(3) 故障修复：

•）一旦确定故障原因，技术支持团队应立即采取措施修复故障。

•）修复措施可能包括修复软件漏洞、恢复数据库、重启服务等。

(4) 故障通知：

•）在故障修复过程中，相关团队应及时向用户和利益相关者通知故障进展和预计的恢复时间。

•）如果需要，可以提供临时解决方案或工作回退计划。

(5) 故障跟踪与记录：

•）所有故障修复过程应详细记录，包括故障描述、诊断过程、修复措施和恢复时间等信息。

•）这些记录可以用于后续的故障分析和改进。

第六部分：验收准则

6.1 功能要求

本系统需要完成之前所列出的所有功能，并且通过相应的标准测试。除了标准的测试，我们也会编写脚本来模拟用户的行为：病人问诊，发送语音、图片、视频，投诉，医生开具处方等等，同时我们管理员会对于平台进行实时监管，确保平台所有数据都可以被正确安全的保存。

6.2 性能要求

6.2.1 响应时间

所谓的“2-5-10 原则”，简单说，就是当用户能够在 2 秒以内得到响应时，会感觉系统的响应很快；当用户在 2-5 秒之间得到响应时，会感觉系统的响应速度还可以；当用户在 5-10 秒以内得到响应时，会感觉系统的响应速度很慢，但是还可以接受；而当用户在超过 10 秒后仍然无法得到响应时，会感觉系统糟透了，或者认为系统已经失去响应，而选择离开这个 Web 站点，或者发起第二次请求。

因此，一个好的系统必须保证每个页面的切换，每个弹框的处理都在短时间内完成。在本系统中，对于响应时间（如表 2）有以下要求：

表 2: 响应时间要求

项目动作	响应时间	说明
首页进入	<2s	用户输入网址按下回车到页面加载完成
问诊信息页面跳转	<2s	用户点击问诊到问诊初始页面加载完成
个人信息页面跳转	<2s	用户点击查询个人信息到个人信息页面加载完成
检测结果页面跳转	<2s	患者点击查询检测结果信息到检测结果信息页面加载完成
投诉处理结果页面跳转	<2s	患者点击查询投诉结果信息到投诉结果信息页面加载完成

6.2.2 更新处理时间

我们的问诊投诉系统总处于一个不断更新的过程中，因此为了满足用户和系统交互的即时性，方便患者、医生和管理员三者的交流，我们需要设置对数据的更新处理时间（如表 3）。

6.3 储存要求

由于问诊投诉系统里用户的信息以及诊断结果等相关信息主要是以数据库文件的形式进行存储。我们需要根据每一张表的数据大小和表的数目估算出所需要的存储空间，并且确保服务器具有足够的存储空间。另外我们同样需要考虑用户既往病历所占有的空间，对于每个用户预分配一块空间进行文件存储。

表 3: 更新时间要求

项目动作	响应时间	说明
患者检测单更新	<2s	检测单实体出结果到用户能查询到检测结果
问诊信息更新	<2s	医生给出问诊结果到患者能查询到问诊结果
聊天信息更新	<2s	用户发送信息到另一方接受到信息
管理员操作	<2s	管理员对患者、医生、药品等信息实现管理

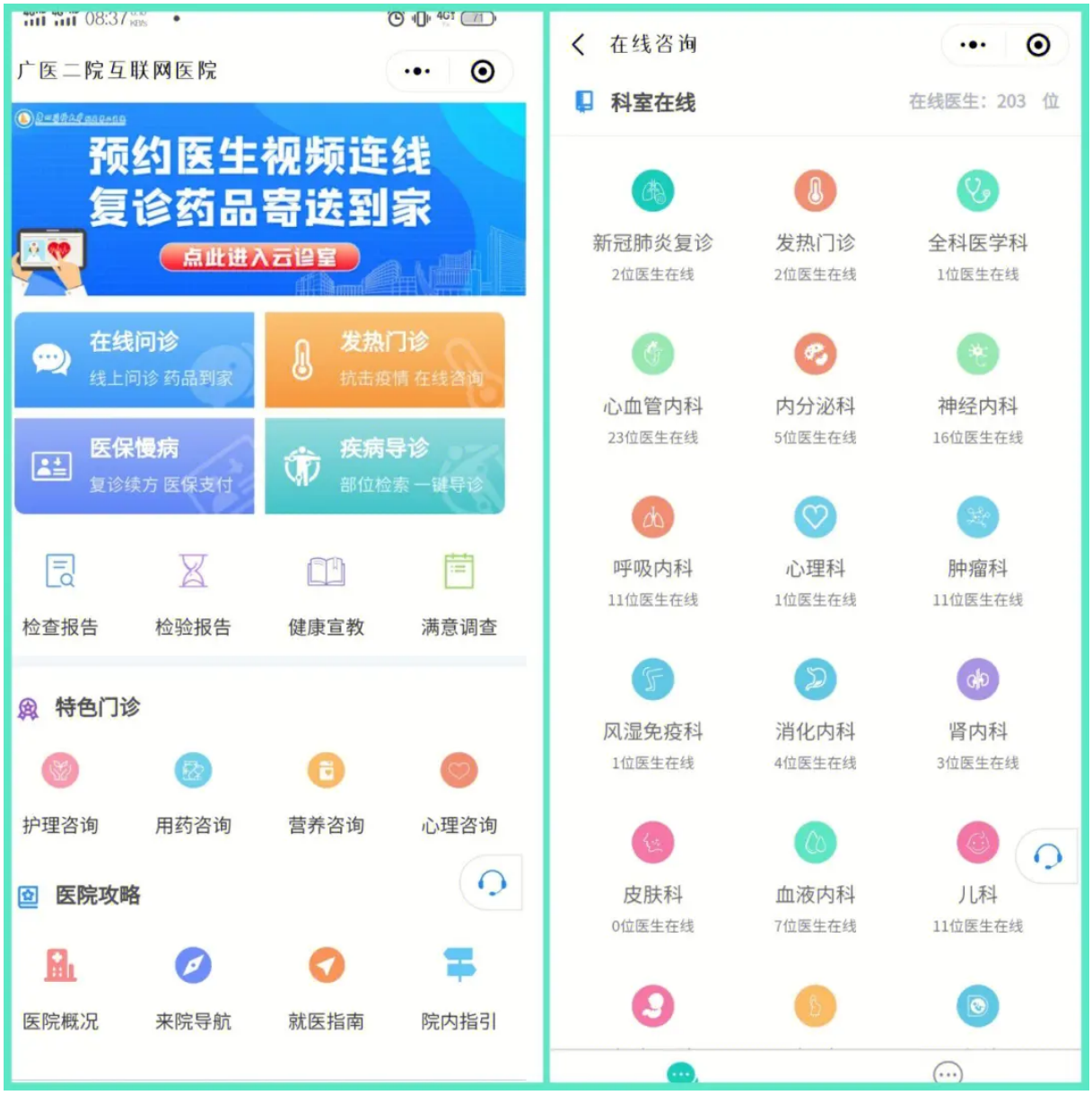
6.4 维护要求

系统开发的过程中程序员必须将记录开发日志，统一开发环境，时刻对源代码进行维护与管理，保证问题可被追踪。软件开发团队需要进行严谨的版本控制，确保开发过程中的软件更新在认为可控的范围之内。并且维护过程中，保证所有用户都无法登录，造成数据的混乱。

第七部分：UI 原型

主要说明问诊投诉子系统的 UI 原型。

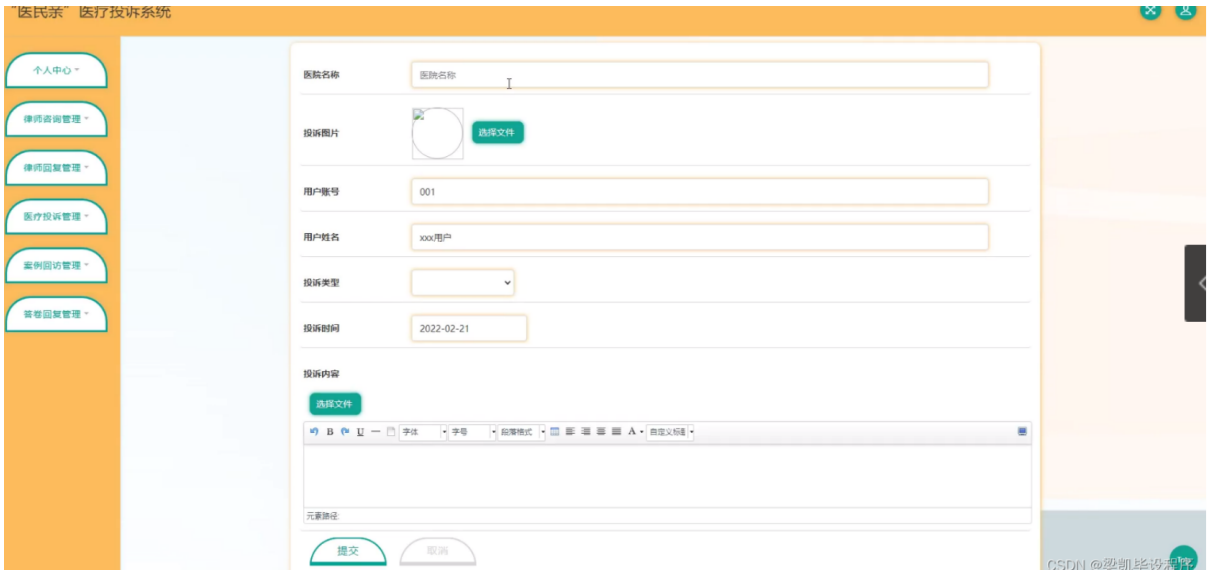
7.1 问诊



mobile 端问诊 UI

web 端问诊 UI

7.2 投诉



患者进行投诉 UI

7.3 管理员管理

管理员管理 UI

7.4 医生进行诊断

医生诊断 UI